Framework for Automating Hardware Verification using LLMs

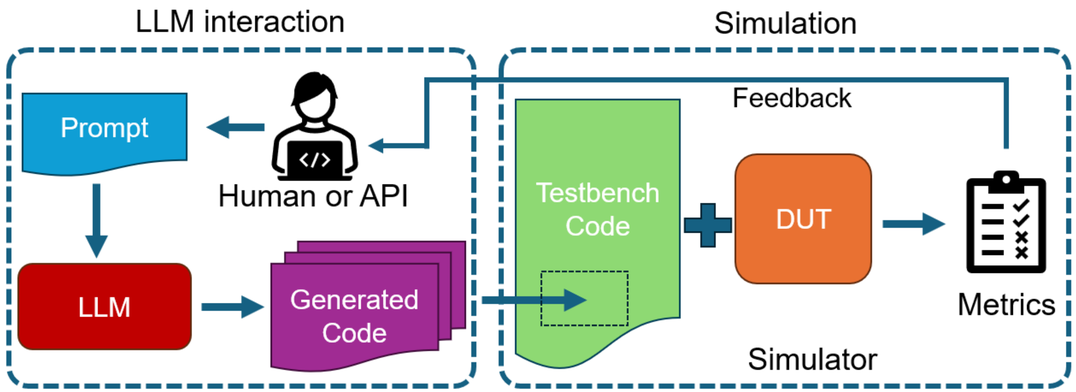
Sean Lowe, Computer Systems Engineering
Mentor: Aman Arora, Assistant Professor
School of Computing and Augmented Computing

Objective & Research Question

LLMs are powerful tools for overcoming the difficulties of digital hardware design and verification. Traditionally, verification is much more time consuming than design - teams typically involve three to five verification engineers per design engineer. We aim to develop an automated framework that can interface with an LLM to generate testbenches and with a hardware simulation tool to evaluate the quality of the generated code. Such a framework can reduce barriers for verification engineers to use LLMs in the industry.

Methods

Our framework is Python-based. We provide prompts to the LLM using its API. GPU-based systems in the ASU's Research Computing cluster are used to test this framework. Currently, the flow supports prompts for generating code in Verilog. It interfaces with a simulator using Makefiles to run simulations and then parses metrics to provide feedback to the LLM about the quality of the generated code.



Conclusion

LLMs can be used for accelerating the digital design verification process. Our framework makes it easy for verification engineers to unlock this potential without having to setup complex tools and write lengthy scripts. Future work involves supporting multiple LLMs, multiple simulation tools, and enabling different LLMs to work together to handle different parts of the verification process.

Acknowledgements

Special thanks to **Dr. Nakul Gopalan** collaborating on this research, **Alma Babbitt and Elias Hilaneh** for testing the framework, and **TSMC and Intel** for the sponsorship that makes this research possible.

Initial Results

Currently, the framework works with Llama 3.1, supports generating testbenches, interfaces with QuestaSim, and parses coverage data to provide feedback to the LLM. The table below shows line coverage obtained for an example design. The hand-coded version is obtained from GitHub.

Method	#	Statement Coverage by Module (%)			
		sha1.v	sha1_core.v	sha1_w_mem.v	Total
Baseline		88.88	95.13	100	95.69
Generate from specification only	1	80	74.3	100	83.51
	2	80	77.08	100	84.94
	3	80	86.11	100	89.6
Generate from specification and synthetic verification plan	1	88.88	86.11	100	91.03
	2	93.33	90.97	100	94.26

FURI

Ira A. Fulton Schools of Engineering
Arizona State University